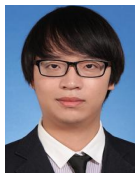
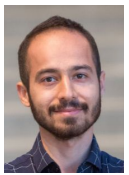
PowerNorm: Rethinking Batch Normalization for Transformers

Sheng Shen, Zhewei Yao, Amir Gholami, Michael Mahoney, Kurt Keutzer

University of California, Berkeley

Executive Summary

- We perform a systematic study of why **Batch Normalization** results in **poor performance in Transformers**
  - Variance in forward pass
  - Variance n backward pass
- We propose **(PN)**, a novel normalization that achieves state-of-the art result in **Machine Translation, Language Modeling**
  - Improve BLEU score on IWSLT by **0.4** and WMT14 by **0.6**
  - Decrease the PPL on PTB by **5.87** and WikiText-103 by **2.78**
- **PN** achieves this by resolving the following problems:
  - Mean estimation (recentering) in BN is better to be removed
  - Better to include running statistics during training
  - The backpropagation needs to be adjusted accordingly

Motivation

- **Normalization is a building block of current neural networks.**

- **There is a lack of interests in NLP for studying normalization**, especially for transformer model (where the training is costly).
  - Layer Normalization. (ICLR' 17)
  - Root Mean Square Layer Normalization (NeurIPS' 19)
  - Understanding and improving layer normalization (NeurIPS' 19)
  - Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention (EMNLP' 19)
  - On Layer Normalization in the Transformer Architecture (ICML' 20)

- **Implementing Batch Normalization directly on NLP does not work.**
  - Why does not it work?
  - How can we make it work on NLP?
  - What can we learn from that?

Content

- **How Normalization is performed in NLP (vs. CV)?**
- Rethinking the variance of Batch Normalization for NLP (Transformers):
  - The variance in forward pass
  - The variance in backward pass
- Our New Normalization PN and its performance in different NLP tasks.

How Normalization is performed in NLP

- **Weight Normalization.**

$$h' = \frac{w^T h}{\|w\|_2}$$

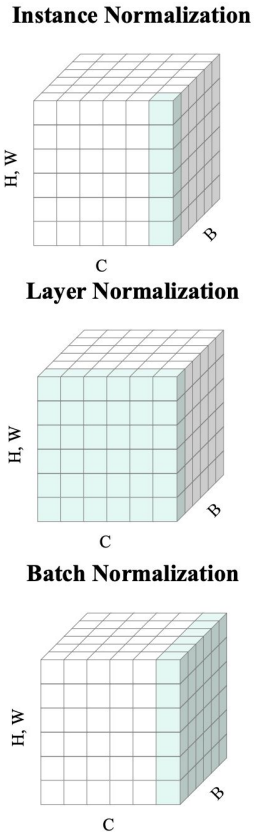
- **Activation Normalization.**

$$\hat{h}_{ncij} = \gamma \frac{h_{ncij} - \Sigma_{k \in \Omega} w_k \mu_k}{\sqrt{\Sigma_{k \in \Omega} w'_k \sigma_k^2 + \epsilon}} + \beta,$$

IN: $\mu_{\text{in}} = \frac{1}{HW} \sum_{i,j}^{H,W} h_{ncij}, \quad \sigma^2_{\text{in}} = \frac{1}{HW} \sum_{i,j}^{H,W} (h_{ncij} - \mu_{\text{in}})^2,$

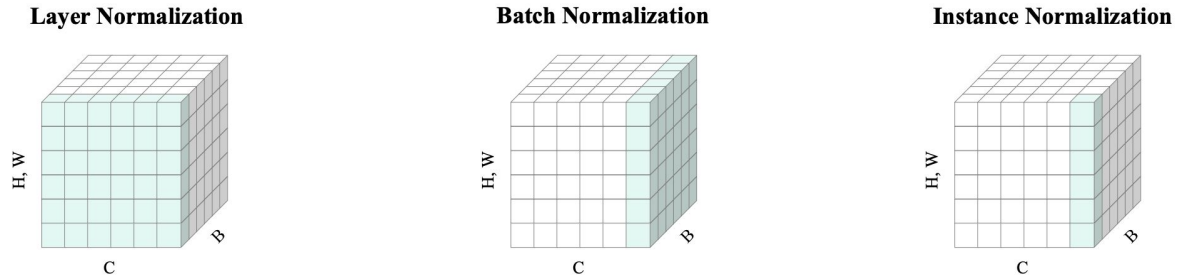
LN: $\mu_{\text{ln}} = \frac{1}{C} \sum_{c=1}^{C} \mu_{\text{in}}, \quad \sigma^2_{\text{ln}} = \frac{1}{C} \sum_{c=1}^{C} (\sigma^2_{\text{in}} + \mu^2_{\text{in}}) - \mu^2_{\text{ln}},$

BN: $\mu_{\text{bn}} = \frac{1}{N} \sum_{n=1}^{N} \mu_{\text{in}}, \quad \sigma^2_{\text{bn}} = \frac{1}{N} \sum_{n=1}^{N} (\sigma^2_{\text{in}} + \mu^2_{\text{in}}) - \mu^2_{\text{bn}},$



How Normalization is performed in NLP

- **Normalization in Computer Vision**
  - **H*W (image size), C (channel/feature dim), B (batch size)**

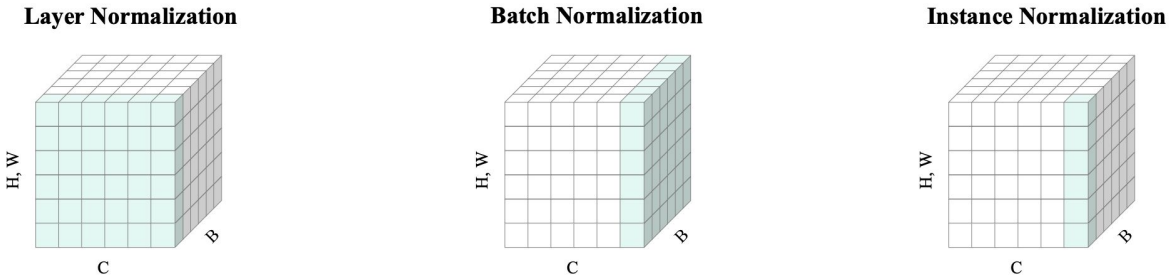


  - **LN:** reduce (C, H*W) => 2*B statistics
  - **BN:** reduce (B, H*W) => 2*C statistics
  - **IN:** reduce (H*W) => 2*B*C statistics

How Normalization is performed in NLP

- **Normalization in NLP**
  - **L/H*W (sentence length), C (feature dim), B (batch size)**

suppose:

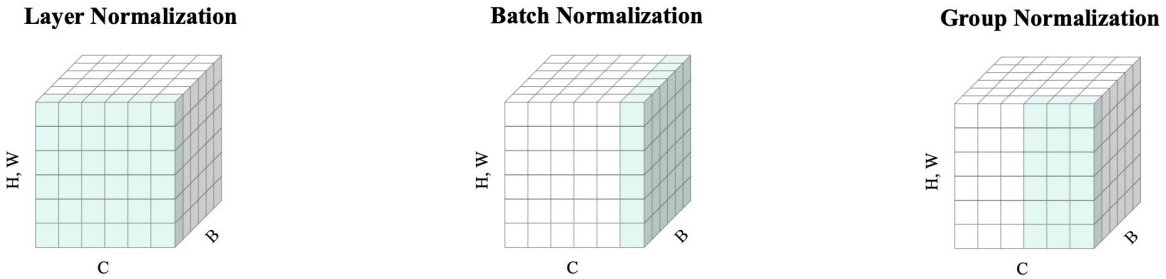


- **LN:** reduce (C, H*W) => 2*B statistics
- **BN:** reduce (B, H*W) => 2*C statistics
- **IN:** reduce (H*W) => 2*B*C statistics

How Normalization is performed in NLP

- **Normalization in NLP**
  - **L/H*W (sentence length), C (feature dim), B (batch size)**

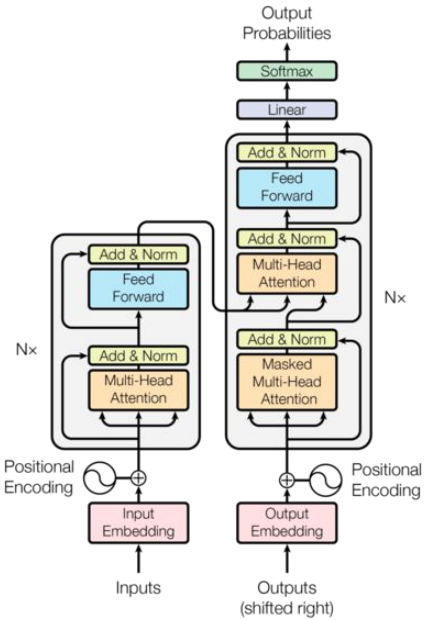
reality:



- **LN:** reduce (C, ~~H*W~~) => 2*B*L statistics
- **BN:** reduce (B, L) => 2*C statistics (padding will destroy the statistics)
- **GN:** reduce (C/g) => 2*B*L*g statistics

How Normalization is performed in NLP

- **Transformer Architecture**



- Encoder: self-attention + point-wise feed-forward

  1 + 2 * N layer norm before/after each block.

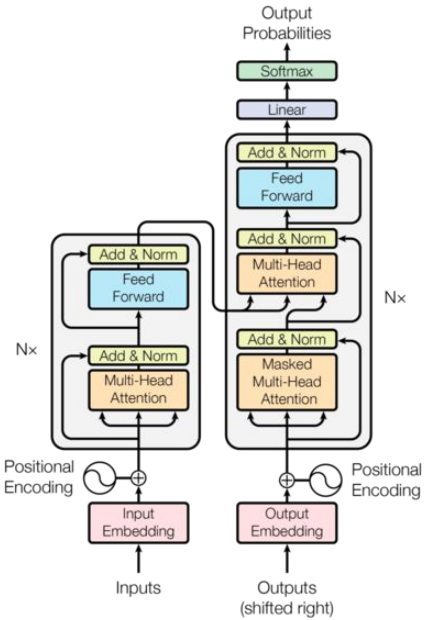
- Decoder: self-attention + enc-decoder attention + point-wise feed-forward

  1 + 3 * N layer norm before/after each block.

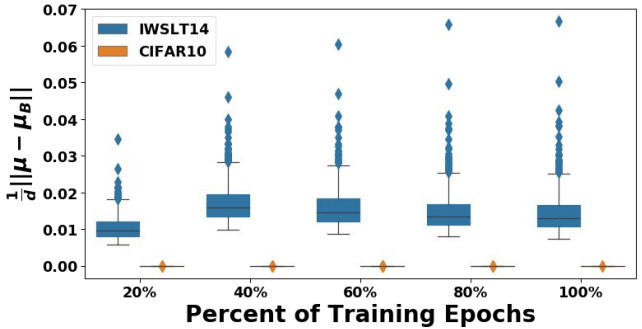
[1] Vaswani et al., “Attention is all you need”, NIPS 2017

How Normalization is performed in NLP

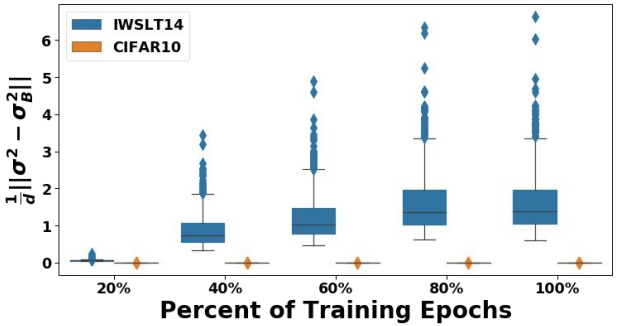
- **Difference btw BN's usage in CNN and Transformer (CV, NLP)**



- **Train/Test Statistical Discrepancy**: the mismatch of running batch-statistics and real batch-statistics for ***NLP***.



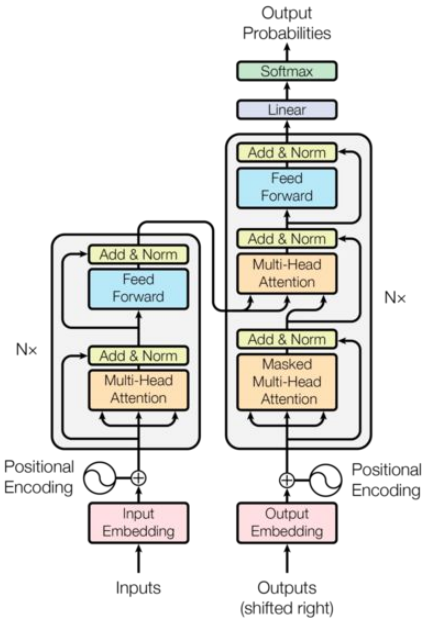
(a) $\|\mu - \mu_B\|^2$ at bn.1 for **CIFAR10** vs **IWSLT** (training)



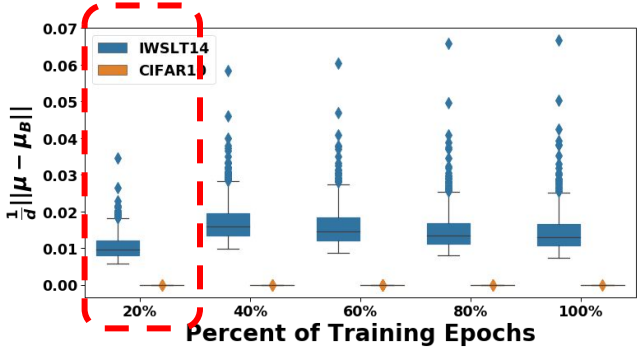
(b) $\|\sigma^2 - \sigma_B^2\|^2$at bn.1 for **CIFAR10** vs **IWSLT** (training)

How Normalization is performed in NLP

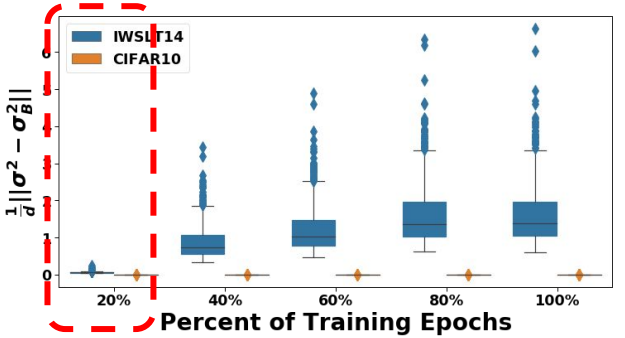
- **Difference btw BN's usage in CNN and Transformer (CV, NLP)**



- **Train/Test Statistical Discrepancy**: the mismatch of running batch-statistics and real batch-statistics for ***NLP***.



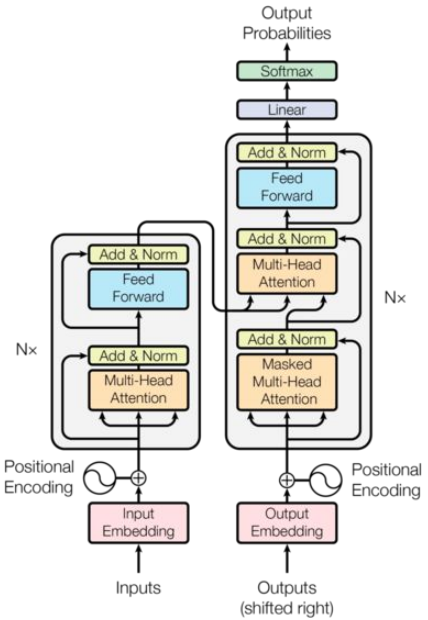
(a) $\|\mu - \mu_B\|^2$ at bn.1 for **CIFAR10** vs **IWSLT** (training)



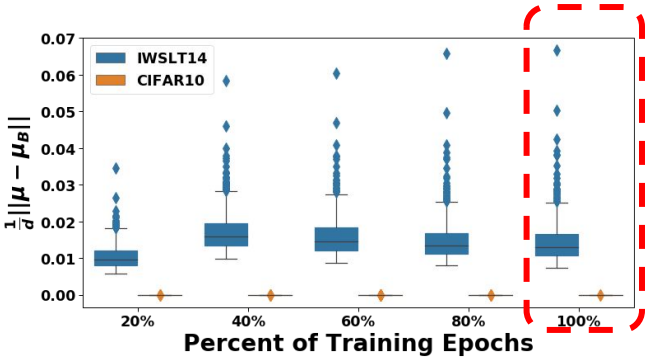
(b) $\|\sigma^2 - \sigma_B^2\|^2$at bn.1 for **CIFAR10** vs **IWSLT** (training)

How Normalization is performed in NLP

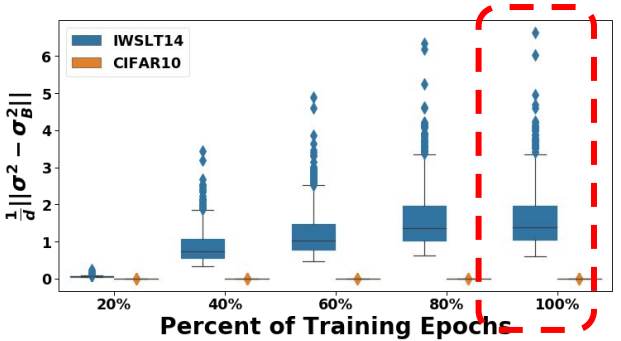
- **Difference btw BN's usage in CNN and Transformer (CV, NLP)**



- **Train/Test Statistical Discrepancy**: the mismatch of running batch-statistics and real batch-statistics for ***NLP***.



(a) $\|\mu - \mu_B\|^2$ at bn.1 for **CIFAR10** vs **IWSLT** (training)



(b) $\|\sigma^2 - \sigma_B^2\|^2$at bn.1 for **CIFAR10** vs **IWSLT** (training)

Content

- How Normalization is performed in NLP (vs. CV)?
- **Rethinking the variance of Batch Normalization for NLP (Transformers):**
  - The variance in forward pass
  - The variance in backward pass
- Our New Normalization PN and its performance in different NLP tasks.

Rethinking the variance of Batch Normalization for NLP

- **The variance in the Forward Pass:**
  - The variance btw the real mean/variance and running mean/variance.
    - $\|\mu - \mu_B\|^2$ and $\|\sigma^2 - \sigma_B^2\|^2$

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

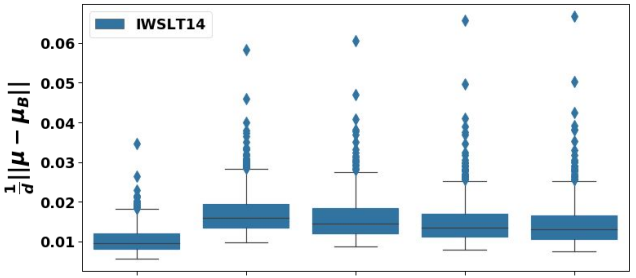
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma^2_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

$$\mu := \mu + \alpha(\mu_{\mathcal{B}} - \mu) \qquad \text{// Update moving averages}$$
$$\sigma := \sigma + \alpha(\sigma_{\mathcal{B}} - \sigma)$$



(a) $\|\mu - \mu_B\|^2$ at bn.1 for **IWSLT** (training)

Rethinking the variance of Batch Normalization for NLP

- **The variance in the Forward Pass:**
  - The variance btw the real mean/variance and running mean/variance.
    - $\|\mu - \mu_B\|^2$ and $\|\sigma^2 - \sigma_B^2\|^2$

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

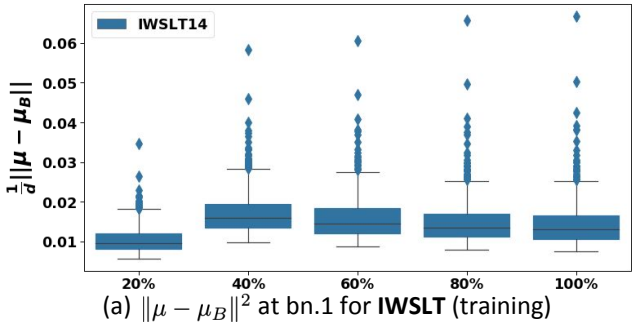
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

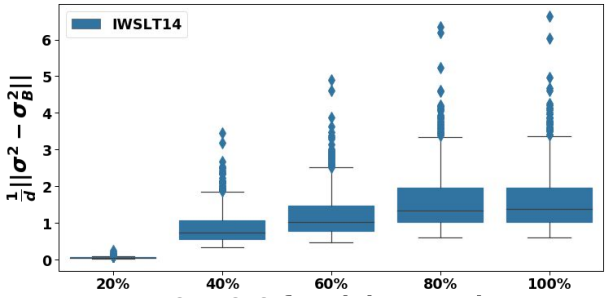
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

$$\mu := \mu + \alpha(\mu_{\mathcal{B}} - \mu) \qquad \text{// Update moving averages}$$
$$\sigma := \sigma + \alpha(\sigma_{\mathcal{B}} - \sigma)$$



(a) $\|\mu - \mu_B\|^2$ at bn.1 for **IWSLT** (training)



(b) $\|\sigma^2 - \sigma_B^2\|^2$ at bn.1 for **IWSLT** (training)

Rethinking the variance of Batch Normalization for NLP

- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - $\mathbf{g}_{\text{mean}}$ and $\mathbf{g}_{\text{var}}$ depend on the statistics of the current mini-batch, which will introduce variance.

$$\mu = \frac{1}{B}\sum_{i=1}^{N} x_i$$

$$\sigma^2 = \frac{1}{B}\sum_{i=1}^{N}(x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta$$

(1) **Forward Pass of BN**

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j} - \hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j\right)$$

$\mathbf{g}_{\text{mean}}$ $\mathbf{g}_{\text{var}}$

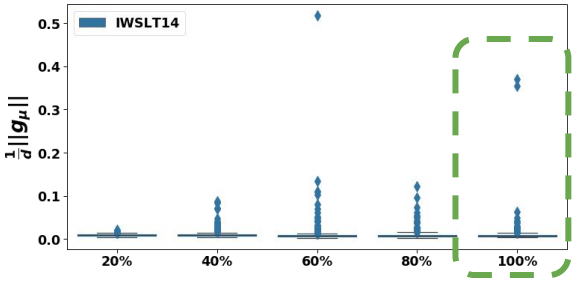
(2) **Backward Pass of BN**

Rethinking the variance of Batch Normalization for NLP

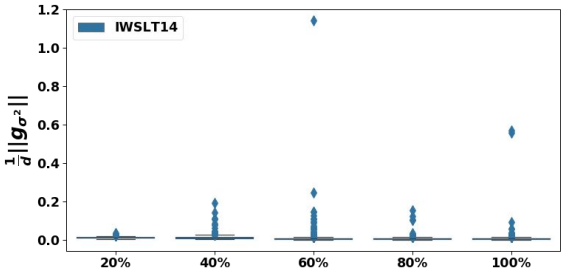
- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - g_{mean} and g_{var} depend on the statistics of the current mini-batch, which will introduce variance.

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \underbrace{\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}}_{g_{mean}} - \underbrace{\hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j}_{g_{var}}\right)$$

Backward Pass of BN



(a) The variance of g_{mean} w.r.t different **IWSLT** batches



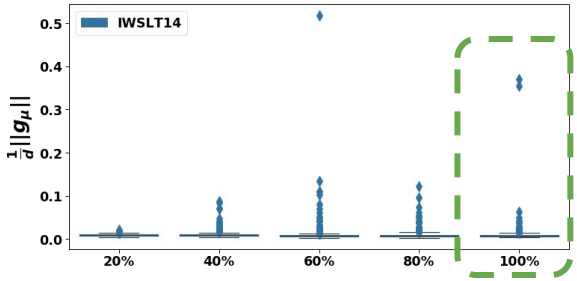
(b) The variance of g_{var} w.r.t different **IWSLT** batches

Rethinking the variance of Batch Normalization for NLP

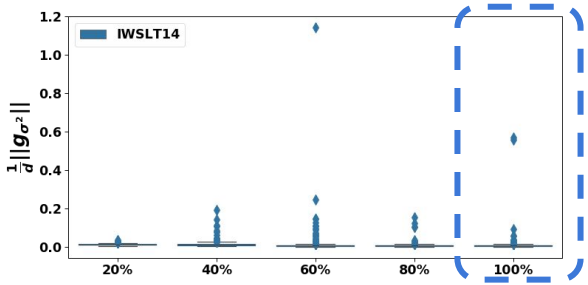
- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - $\mathbf{g}_{\text{mean}}$ and $\mathbf{g}_{\text{var}}$ depend on the statistics of the current mini-batch, which will introduce variance.

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \underbrace{\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}}_{\mathbf{g}_{\text{mean}}} - \underbrace{\hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j}_{\mathbf{g}_{\text{var}}}\right)$$

Backward Pass of BN



(a) The variance of $\mathbf{g}_{\text{mean}}$ w.r.t different **IWSLT** batches



(b) The variance of $\mathbf{g}_{\text{var}}$ w.r.t different **IWSLT** batches

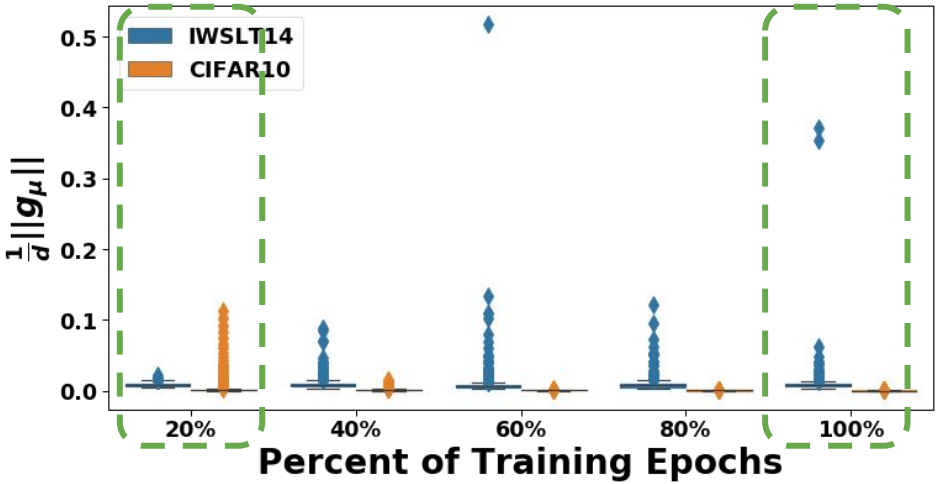
Rethinking the variance of Batch Normalization for NLP

- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - **CV** vs. **NLP**.

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j} - \hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j\right)$$

g_{mean}

Backward Pass of BN



(a) The variance of g_{mean} w.r.t different **CIFAR10** vs **IWSLT** batches

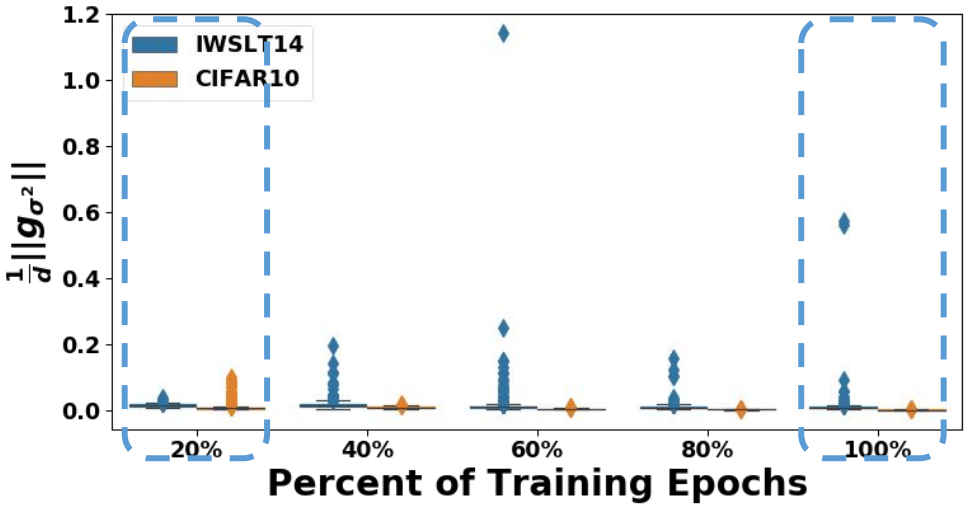
Rethinking the variance of Batch Normalization for NLP

- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - **CV** vs. **NLP**.

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j} - \hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j\right)$$

g_{var}

Backward Pass of BN



(b) The variance of g_{var} w.r.t different **CIFAR10** vs **IWSLT** batches

Content

- How Normalization is performed in NLP (vs. CV)?
- Rethinking Batch Normalization for NLP (Transformers):
  - The variance in forward pass
  - The variance in backward pass
- **Our New Normalization PN and its performance in different NLP tasks.**

PN (Power Normalization)

1. **Forward Pass: PN-V (Variance-reduced Batch Normalization)**

   1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.
   1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

2. **Backward Pass: PN**

   2.1 Replace real $(x_B)^2$ with running $(x)^2$ while training
   2.2 Stabilizing the backward pass with accumulating gradient by Exponential Moving Average approximation.

PN (Power Normalization)

1. Forward Pass: PN-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi_B}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

(1.1) **Forward Pass of PN-V**

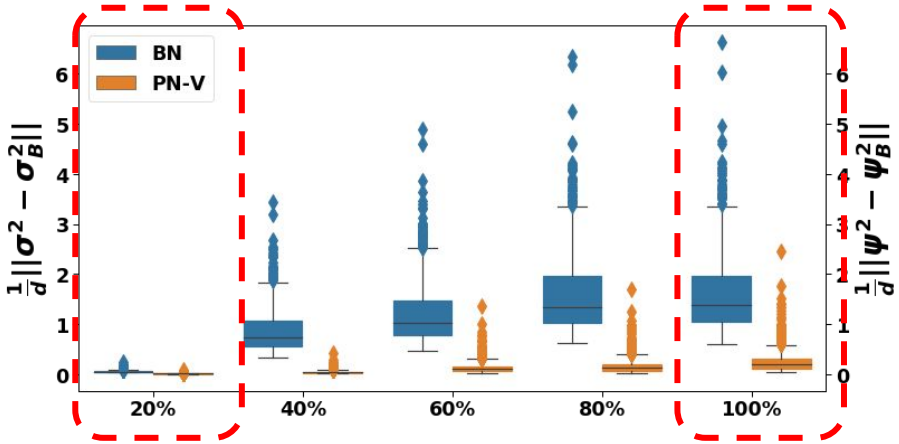
$$\mu = \frac{1}{B}\sum_{i=1}^{N} x_i$$

$$\sigma^2 = \frac{1}{B}\sum_{i=1}^{N}(x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta$$

(1.2) **Forward Pass of BN**



(a) $\|\sigma^2 - \sigma_B^2\|^2$ vs. $\|\phi^2 - \phi_B^2\|^2$ at bn.1 w.r.t **IWSLT** (training)

PN (Power Normalization)

1. Forward Pass: PN-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial x_i} &= \frac{\partial \mathcal{L}}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial x_i} + \sum_{j\in B}\frac{\partial \mathcal{L}}{\partial \hat{x}_j}\frac{\partial \hat{x}_j}{\partial {\psi_B}^2}\frac{\partial {\psi_B}^2}{\partial x_i} \\ &= \frac{\partial \mathcal{L}}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial x_i} + \sum_{j\in B}\frac{\partial \mathcal{L}}{\partial \hat{x}_j}(-\frac{1}{2}\frac{x_j}{{\psi_B}^3})\frac{2x_i}{B} \\ &= \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{x}_i} - \frac{1}{B\psi_B}\sum_{j\in B}\frac{\partial \mathcal{L}}{\partial \hat{x}_j}\hat{x}_j\hat{x}_i.\end{aligned}$$

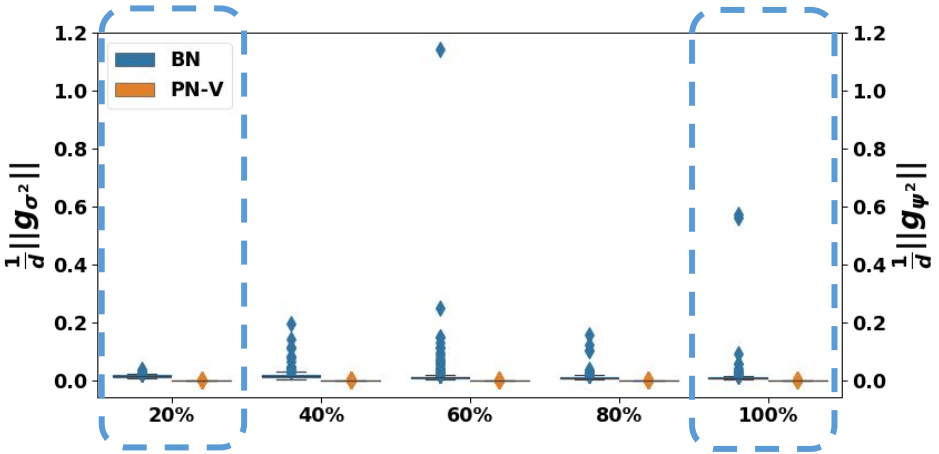
g_{phi}

(2.1) **Backward Pass of PN-V**

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\Big(B\frac{\partial L}{\partial \hat{x}_i} - \sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j} - \hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j\Big)$$

g_{mean} g_{var}

(2.2) **Backward Pass of BN**



(b) The variance of g_{var} vs g_{phi} w.r.t different **IWSLT** batches

PN (Power Normalization)

1. Forward Pass: PN-V (Variance-reduced Batch Normalization)

1.2 Replace the running variance $(x-\mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\psi^2 = \psi^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

(1) **Running Statistics of PN-V**

$$\mu_B = \frac{1}{B}\sum_{i=1}^{B} x_i$$

$$\sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(x_i - \mu)^2$$

$$\mu = \mu + \alpha(\mu_B - \mu) \quad \text{update moving averages}$$

$$\sigma = \sigma + \alpha(\sigma_B - \sigma) \quad \text{update moving averages}$$

(2) **Running Statistics of BN**

- The EMA (Exponential Moving Average) estimation of variance is not a desirable estimation. And it is hard to make it correct[1];

$$\begin{aligned}\sigma_t^2 &= \alpha\sigma_{t-1}^2 + (1-\alpha)(x_t - \mu_t)(x_t - \mu_{t-1}) \\ &= (1-\alpha)(\sigma_{t-1}^2 + \alpha(x_t - \mu_{t-1})^2)\end{aligned}$$

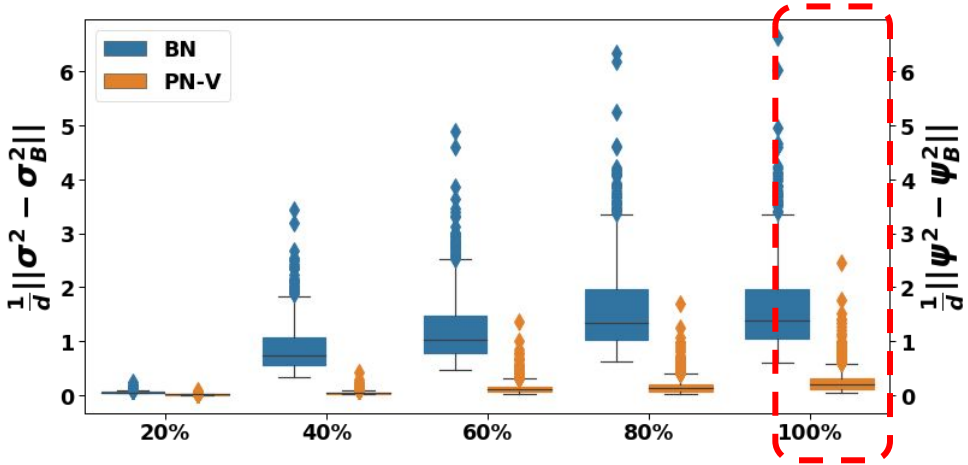
[1] Tony Finch, Incremental calculation of weighted mean and variance, Lecture Note;

PN (Power Normalization)

2. Further correct the forward and backward: PN

2.1 Replace real $(x_B)^2$ with running $(x)^2$ while training (forward pass)

2.2 Stabilizing the backward pass with accumulating gradient by Exponential Moving Average approximation.



$\|\phi^2 - \phi_B^2\|^2$ at bn.1 for IWSLT (training)

PN (Power Normalization)

2. Further correct the forward and backward: PN

2.1 Replace real $(x_B)^2$ with running $(x)^2$ while training (forward pass)

2.2 Stabilizing the backward pass with accumulating gradient by Exponential Moving Average approximation.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

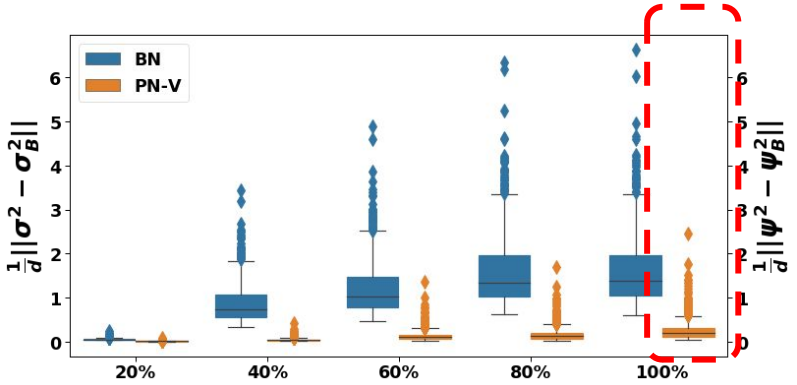
(1) **Forward Pass of PN**

$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{x}_i} - \underbrace{\frac{1}{B\psi_B}\sum_{j\in B}\frac{\partial \mathcal{L}}{\partial \hat{x}_j}\hat{x}_j\hat{x}_i}_{\tilde{x}'_B}.$$

$$\tilde{x}'_B = \hat{x}'_B - \tau_T^{\hat{x}} \odot \hat{x}_B$$

$$\tau_T^{\hat{x}} = \tau_{T-1}^{\hat{x}} \odot [1 - (1-\alpha)\cdot \hat{x}_B \odot \hat{x}_B] + (1-\alpha)\hat{x}'_B \odot \hat{x}_B$$

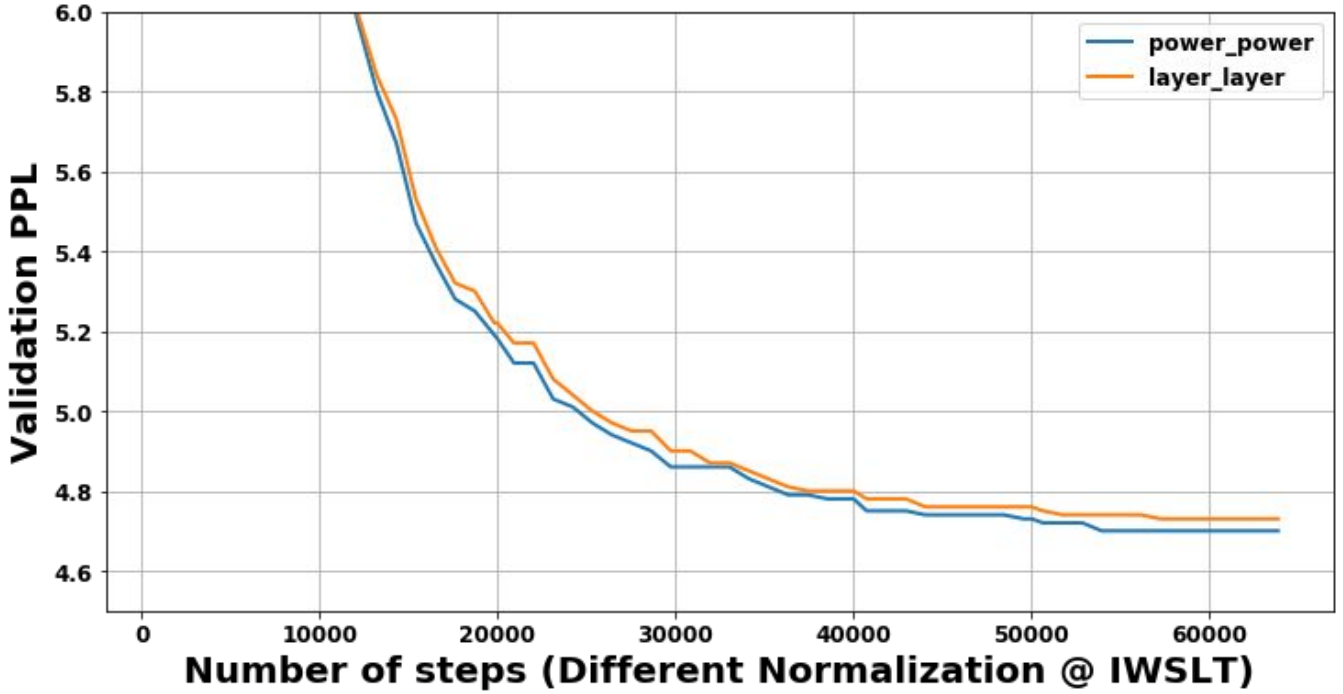
(2) **Backward Pass of PN**



$\|\phi^2 - \phi_B^2\|^2$ at bn.1 for IWSLT (training)

PN (Power Normalization) for MT

- IWSLT14 De-En (0.16M translation pairs) Training Curve. (PN vs. LN)



PN (Power Normalization) for MT

- IWSLT14 De-En (0.16M translation pairs)

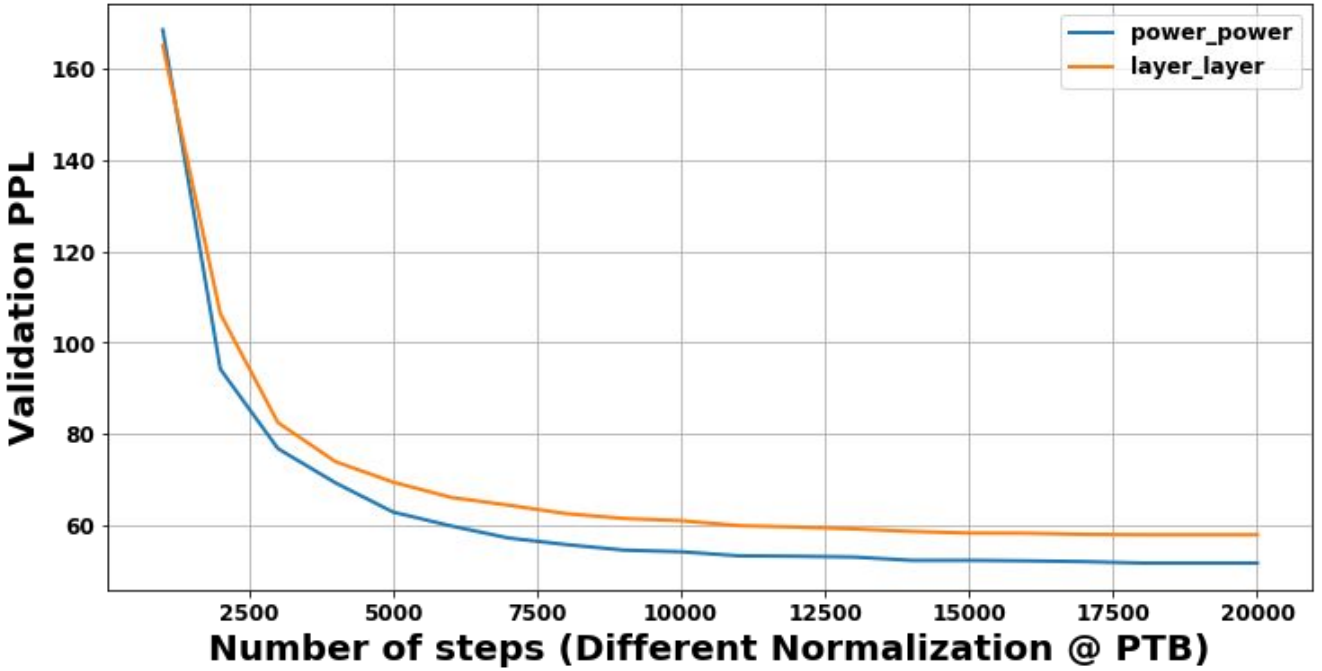
Small (36.7M Param) L6+L6 D512 H4	BLEU
Layer Norm	35.5
Batch Norm	**34.4**
PN-V (Ours)	35.4
PN (Ours)	**35.9**

- WMT14 En-De (4.5M translation pairs)

Big (68.2M Param) L6+L6 D1024 H16	BLEU
Layer Norm	29.5
Batch Norm	**28.1**
PN-V (Ours)	28.5
PN (Ours)	**30.1**

PN (Power Normalization) for LM

- **PennTree Bank (0.09M tokens)**
  - Training Curve. (PN vs. LN)



PN (Power Normalization) for LM

- **PennTree Bank (0.09M tokens)**

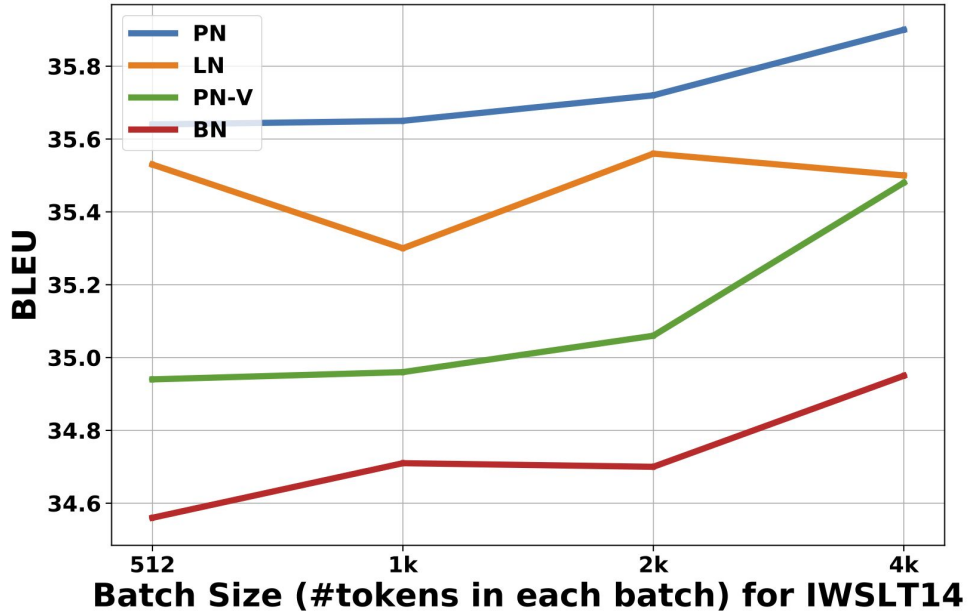
BDP-1Core (12.0M Param)	PPL (Lower is better)
Layer Norm	53.19
Batch Norm	**64.72**
PN (Ours)	**47.32**

- **Wikitext-103 (103M tokens)**

BDP-1Core (85.3M Param)	PPL (Lower is better)
Layer Norm	20.90
Batch Norm	**27.01**
PN (Ours)	**18.12**

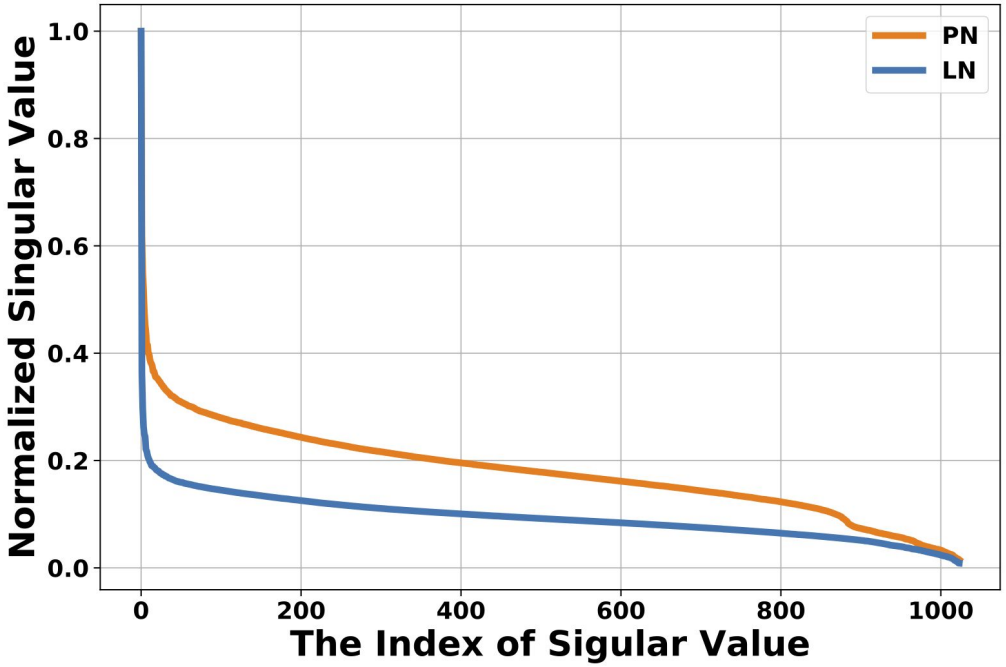
Analysis (various batch sizes)

- PN is the most robust in various batch-size setting.



Analysis (embedding layer)

- PN leads to a more well-conditioned embedding layer v.s. LN.



Conclusion

- The variance between batch statistics/running statistics results in the poor performance of **Batch Normalization in Transformers**

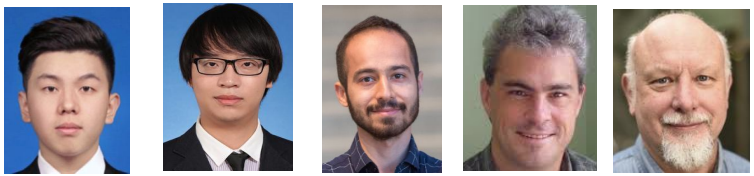
- Reduce the variance through **PN** can achieves significantly better result in **Machine Translation, Language Modelling**
  - Improve BLEU score on IWSLT by **0.4** and WMT14 by **0.6**
  - Decrease the PPL on PTB by **5.87** and Wiki-Text by **2.78**

Reference

Thank you for you attention

Paper available at https://arxiv.org/abs/2003.07845.

Code available at https://github.com/sIncerass/powernorm/



E^{2}N (effective and efficient normalization)

1. Forward Pass: E^{2}N-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi_B}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

(1) **Forward Pass of PN-V**

$$\mu = \frac{1}{B}\sum_{i=1}^{N} x_i$$

$$\sigma^2 = \frac{1}{B}\sum_{i=1}^{N} (x_i - \mu)^2$$

$$\hat{x}_i = \frac{x_i - \mu}{\sqrt{\sigma^2 + \epsilon}}$$

$$y_i = \gamma \hat{x}_i + \beta$$

(1) **Forward Pass of BN**

$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{\partial \mathcal{L}}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{x}_j}\frac{\partial \hat{x}_j}{\partial {\psi_B}^2}\frac{\partial {\psi_B}^2}{\partial x_i}$$

$$= \frac{\partial \mathcal{L}}{\partial \hat{x}_i}\frac{\partial \hat{x}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{x}_j}\left(-\frac{1}{2}\frac{x_j}{{\psi_B}^3}\right)\frac{2x_i}{B}$$

$$= \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{x}_i} - \frac{1}{B\psi_B}\sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{x}_j}\hat{x}_j\hat{x}_i.$$

g_{phi}

(2) **Backward Pass of PN-V**

E^2N (effective and efficient normalization)

1. Forward Pass: E^2N-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi_B}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

(1) **Forward Pass of E^2N-V**

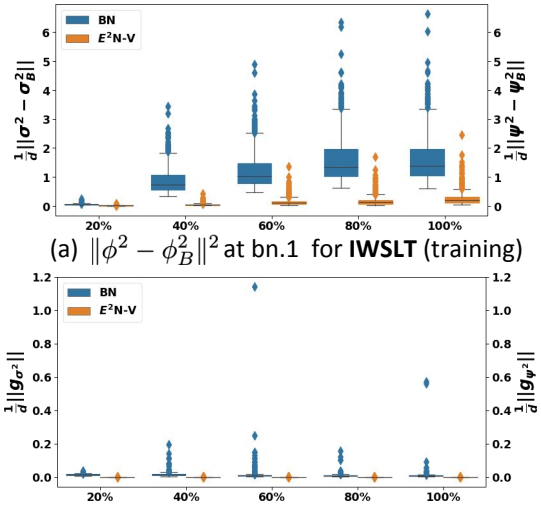
$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_i}\frac{\partial \hat{\mathrm{x}}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_j}\frac{\partial \hat{\mathrm{x}}_j}{\partial {\psi_B}^2}\frac{\partial {\psi_B}^2}{\partial x_i}$$

$$= \frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_i}\frac{\partial \hat{\mathrm{x}}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_j}\left(-\frac{1}{2}\frac{x_j}{{\psi_B}^3}\right)\frac{2x_i}{B}$$

$$= \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_i} - \frac{1}{B\psi_B}\sum_{j \in B}\frac{\partial \mathcal{L}}{\partial \hat{\mathrm{x}}_j}\hat{\mathrm{x}}_j\hat{\mathrm{x}}_i.$$

g_{phi}

(2) **Backward Pass of E^2N-V**



(a) $\|\phi^2 - \phi_B^2\|^2$ at bn.1 for **IWSLT** (training)

(b) The variance of g_{phi} w.r.t different **IWSLT** batches

E^2N (effective and efficient normalization)

1. Forward Pass: E^2N-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

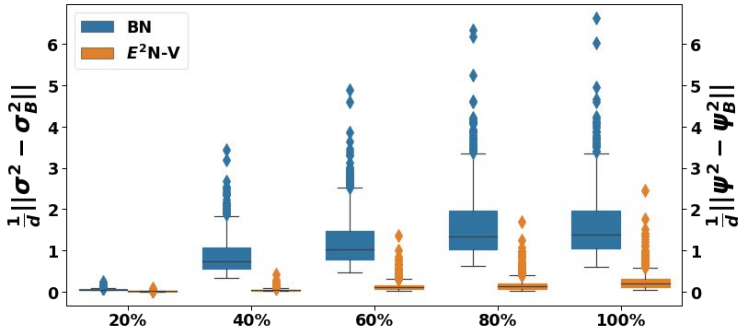
1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(x_i - \mu_B)^2$$

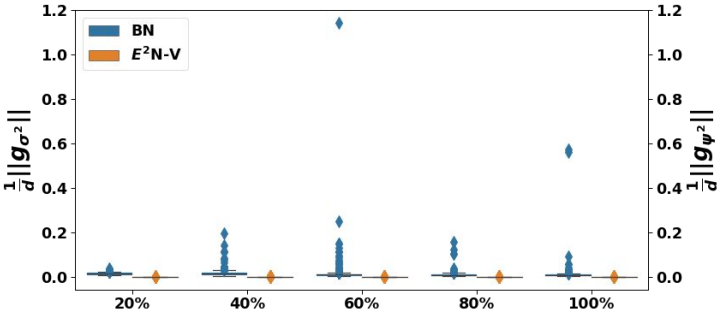
(1) Forward Pass of BN

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

(2) Forward Pass of E^2N-V



(a) $\|\sigma^2 - \sigma_B^2\|^2$ vs. $\|\phi^2 - \phi_B^2\|^2$ at bn.1 w.r.t **IWSLT** (training)



(b) The variance of g_{var} vs g_{phi} w.r.t different **IWSLT** batches

E²N (effective and efficient normalization)

2. Further correct the forward and backward: E²N

2.1 Replace real $(x_B)^2$ with running $(x)^2$ while training (forward pass)

2.2 Stabilizing the backward pass with accumulating gradient by Exponential Moving Average approximation.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

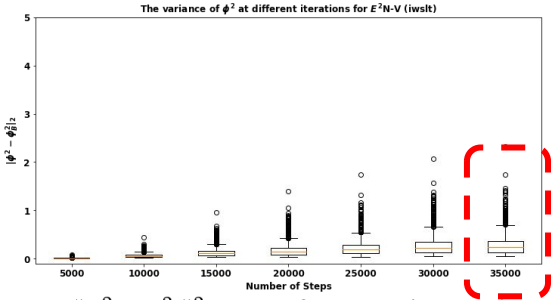
(1) **Forward Pass of E²N**

$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_i} - \underbrace{\frac{1}{B\psi_B}\sum_{j\in B}\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_j}\hat{\mathbf{x}}_j}_{\tilde{x}'_B}\hat{\mathbf{x}}_i.$$

$$\tilde{x}'_B = \hat{x}'_B - \tau_T^{\hat{x}} \odot \hat{x}_B$$

$$\tau_T^{\hat{x}} = \tau_{T-1}^{\hat{x}} \odot [1 - (1-\alpha)\cdot \hat{x}_B \odot \hat{x}_B] + (1-\alpha)\hat{x}'_B \odot \hat{x}_B$$

(2) **Backward Pass of E²N**



$\|\phi^2 - \phi_B^2\|^2$ at bn.1 for IWSLT (training)

E²N (effective and efficient normalization)

1. Forward Pass: E²N-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

$$\hat{x}_i = \frac{x_i}{\psi_B}$$

$$y_i = \gamma \hat{x}_i + \beta$$

$$\psi^2 = \psi_B^2 + \alpha(\psi_B^2 - \psi^2)$$

// update moving averages

(1) **Forward Pass of E²N-V**

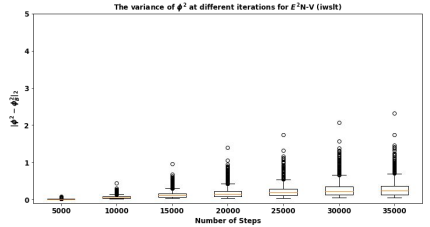
$$\frac{\partial \mathcal{L}}{\partial x_i} = \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_i}\frac{\partial \hat{\mathbf{x}}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_j}\frac{\partial \hat{\mathbf{x}}_j}{\partial {\psi_B}^2}\frac{\partial {\psi_B}^2}{\partial x_i}$$

$$= \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_i}\frac{\partial \hat{\mathbf{x}}_i}{\partial x_i} + \sum_{j \in B} \frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_j}\left(-\frac{1}{2}\frac{x_j}{{\psi_B}^3}\right)\frac{2x_i}{B}$$

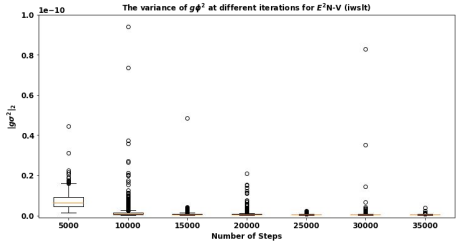
$$= \frac{1}{\psi_B}\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_i} - \frac{1}{B\psi_B}\sum_{j \in B}\frac{\partial \mathcal{L}}{\partial \hat{\mathbf{x}}_j}\hat{\mathbf{x}}_j\hat{\mathbf{x}}_i.$$

g_{phi}

(2) **Backward Pass of E²N-V**



(a) $\|\phi^2 - \phi_B^2\|^2$ at bn.1 for **IWSLT** (training)



(b) The variance of g_{phi} w.r.t different **IWSLT** batches

Rethinking the variance of Batch Normalization for NLP

- **The variance in the Forward Pass:**
  - The variance btw the real mean/variance and running mean/variance.
    - $\|\mu - \mu_B\|^2$ and $\|\sigma^2 - \sigma_B^2\|^2$

Input: Values of x over a mini-batch: $\mathcal{B} = \{x_{1...m}\}$;
Parameters to be learned: γ, β
Output: $\{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\}$

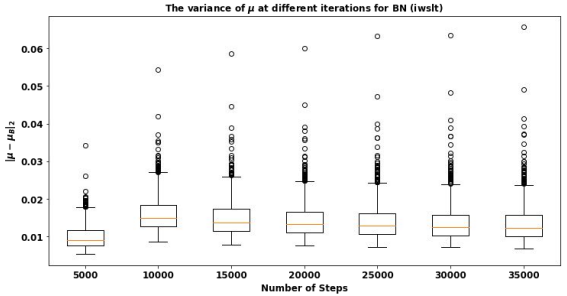
$$\mu_{\mathcal{B}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \qquad \text{// mini-batch mean}$$

$$\sigma_{\mathcal{B}}^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \qquad \text{// mini-batch variance}$$

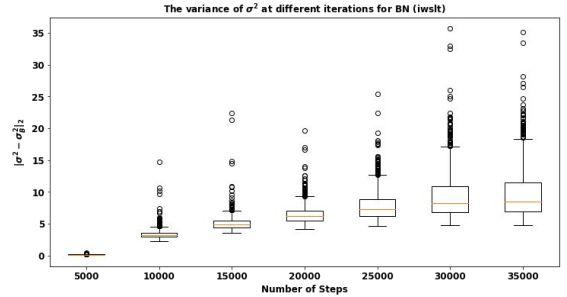
$$\widehat{x}_i \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma_{\mathcal{B}}^2 + \epsilon}} \qquad \text{// normalize}$$

$$y_i \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i) \qquad \text{// scale and shift}$$

$$\mu := \mu + \alpha(\mu_{\mathcal{B}} - \mu) \qquad \text{// Update moving averages}$$
$$\sigma := \sigma + \alpha(\sigma_{\mathcal{B}} - \sigma)$$



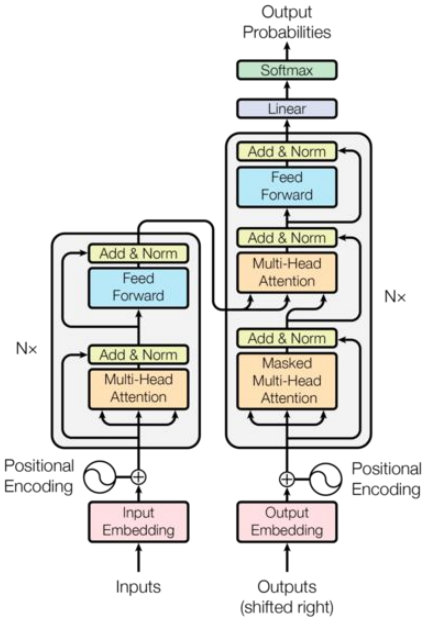
(a) $\|\mu - \mu_B\|^2$ at bn.1 for **IWSLT** (training)



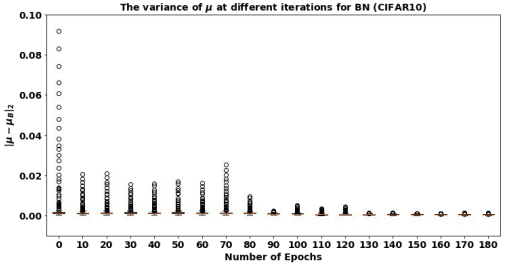
(b) $\|\sigma^2 - \sigma_B^2\|^2$ at bn.1 for **IWSLT** (training)

How Normalization is performed in NLP

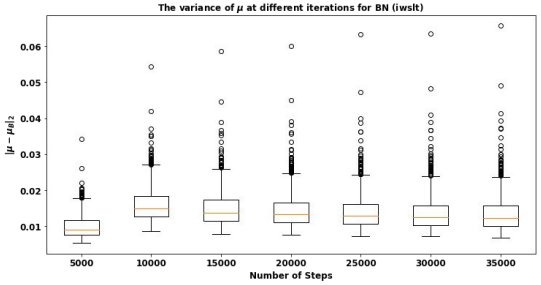
- **Difference btw CNN and Transformer (CV, NLP)**



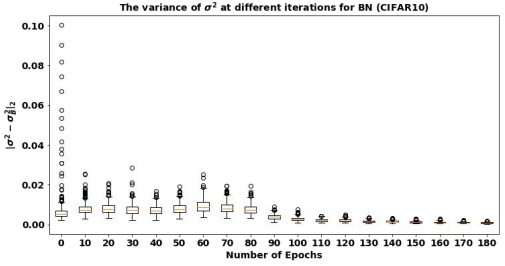
- **Train/Test Statistical Discrepancy**: the mismatch of **running statistics** and **real statistics** in different mini-batches.



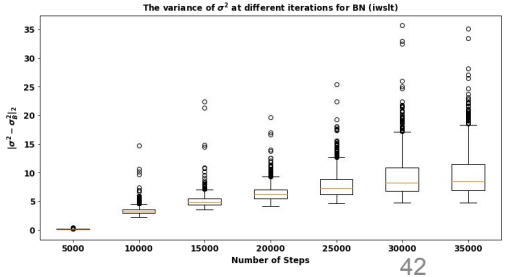
(a.1) $\|\mu - \mu_B\|^2$ at bn.1 for CIFAR10 (training)



(a.2) $\|\mu - \mu_B\|^2$ at bn.1 for **IWSLT** (training)



(b.1) $\|\sigma^2 - \sigma_B^2\|^2$ at bn.1 for CIFAR10 (training)



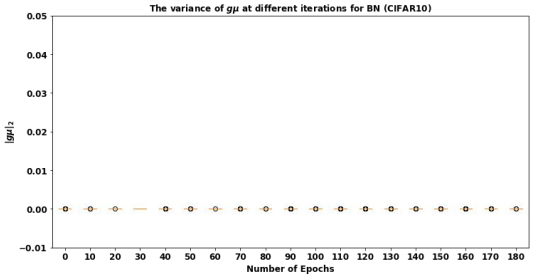
(b.2) $\|\sigma^2 - \sigma_B^2\|^2$ at bn.1 for **IWSLT** (training)

Rethinking the variance of Batch Normalization for NLP

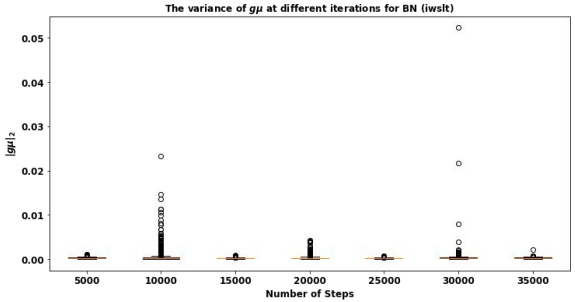
- **The variance in the Backward Pass:**
  - The variance of gradient w.r.t different portion of the data.
    - **CV** vs. **NLP**.

$$\frac{\partial L}{\partial x_i} = \frac{1}{B\sqrt{\sigma^2+\epsilon}}\left(B\frac{\partial L}{\partial \hat{x}_i} - \underbrace{\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}}_{g_{mean}} - \underbrace{\hat{x}_i\sum_{j=1}^{N}\frac{\partial L}{\partial \hat{x}_j}\cdot\hat{x}_j}_{g_{var}}\right)$$

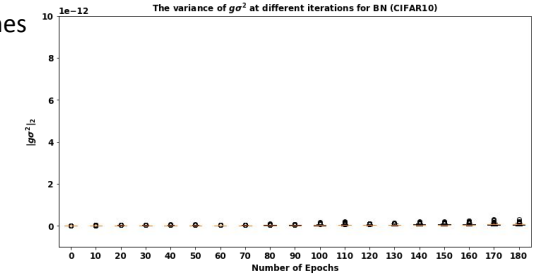
Backward Pass of BN



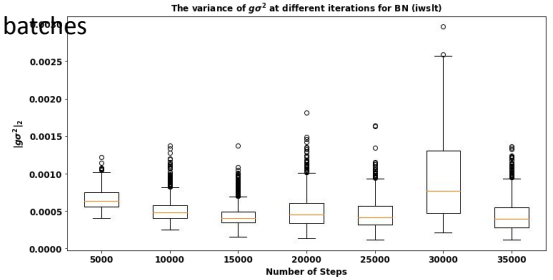
(a.1) The variance of g_{mean} w.r.t different **CIFAR10** batches



(a.2) The variance of g_{mean} w.r.t different **IWSLT** batches



(b.1) The variance of g_{var} w.r.t different **CIFAR10** batches



(b.2) The variance of g_{var} w.r.t different **IWSLT** batches

E²N (effective and efficient normalization)

1. Forward Pass: E²N-V (Variance-reduced Batch Normalization)

1.1 Remove the recentering $(x - \mu_B)$ in the batch normalization.

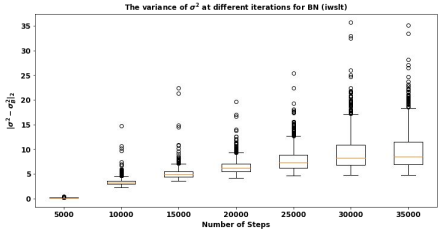
1.2 Replace the running variance $(x - \mu_B)^2$ with more accurate running $(x)^2$ while training.

$$\sigma_B^2 = \frac{1}{B}\sum_{i=1}^{B}(x_i - \mu_B)^2$$

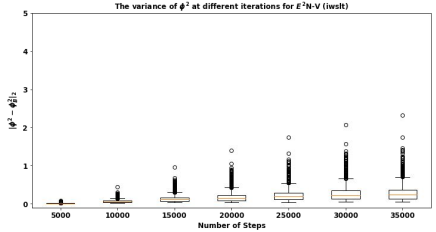
(1) Forward Pass of BN

$$\psi_B^2 = \frac{1}{B}\sum_{i=1}^{B} x_i^2$$

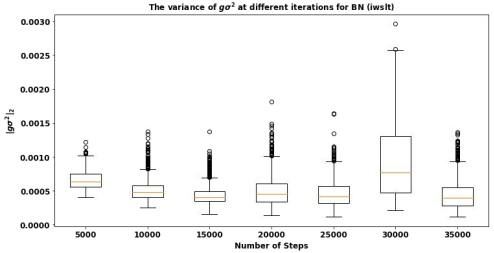
(2) Forward Pass of E²N-V



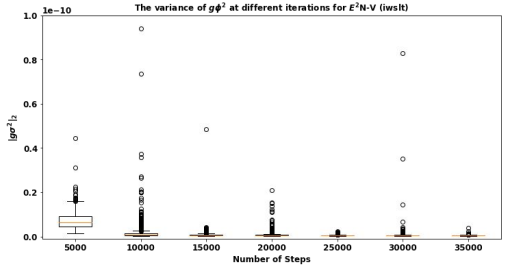
(a.1) $\|\sigma^2 - \sigma_B^2\|^2$ at bn.1 for **IWSLT** (training)



(a.2) $\|\phi^2 - \phi_B^2\|^2$ at bn.1 for **IWSLT** (training)

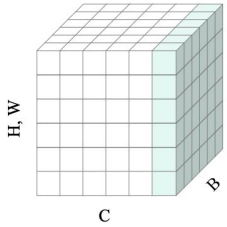


(b.1) The variance of g_{var} w.r.t different **IWSLT**

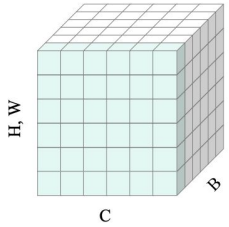


(b.2) The variance of g_{phi} w.r.t different **IWSLT** batches

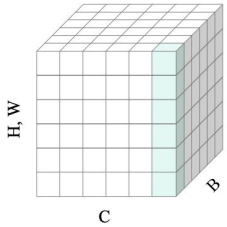
Batch Normalization



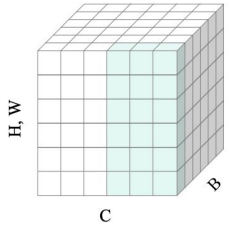
Layer Normalization



Instance Normalization



Group Normalization



E^2N (effective and efficient normalization) for MT

- **IWSLT14 De-En (0.16M translation pairs)**
  - Computation Overhead. (E^2N vs. LN) on one RTX 2080.

Small (36.7M Param) L6+L6 D512 H4	**Inference Speed (wps, word per second)**	**FLOPs**
Layer Norm	3586.843	-
E^2N (Ours)	3697.686	-

E^2N (effective and efficient normalization) for BERT

- **GLEU Results**

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

Small (36.7M Param) L6+L6 D512 H4	**BLEU**
Layer Norm + Deep init[2]	35.63
Layer Norm	34.95
Group Norm	**35.27**
Batch Norm	diverge
Mask Batch Norm	34.51 (w/ rm)[1] 33.79
Mask Instance Norm	diverge

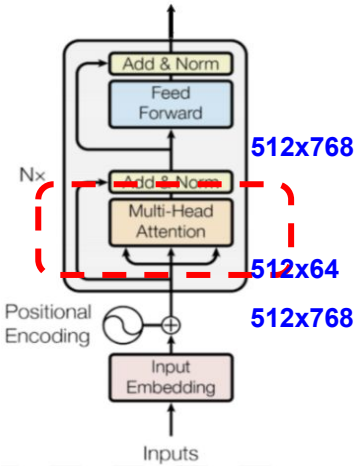
Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Group Norm	**36.05**
Batch Norm	diverge
Mask Batch Norm	35.48 (w/ rm) 34.38
Mask Instance Norm	diverge

[1] w/ rm: using the running mean and running variance while doing the inference.
[2] Zhang et al, Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention, ACL'19

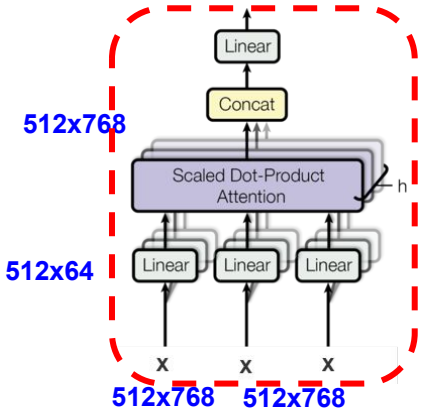
Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why GN ?
  1. Real matrix multiplication in head-level.

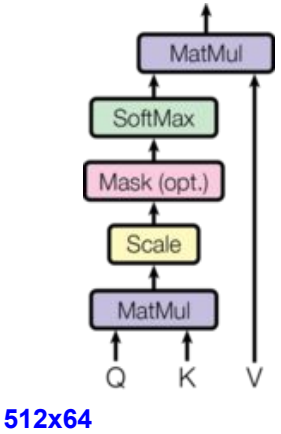


(a) BERT Encoder [Transformer]



(b) Multi-head self-attention (head number: h)

$$\text{MultiHead}(X, X, X) = \text{Concat}(\text{head}_1, ..., \text{head}_h)\, W^O$$
$$\text{where head}_i = \text{Attention}(XW^Q_i, XW^K_i, XW^V_i)$$



(c) Single-head self-attention (hidden size: d_k)

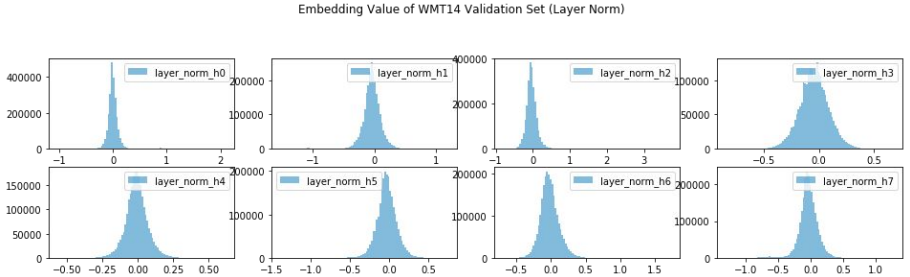
$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V$$

Normalization Results on Neural Machine Translation

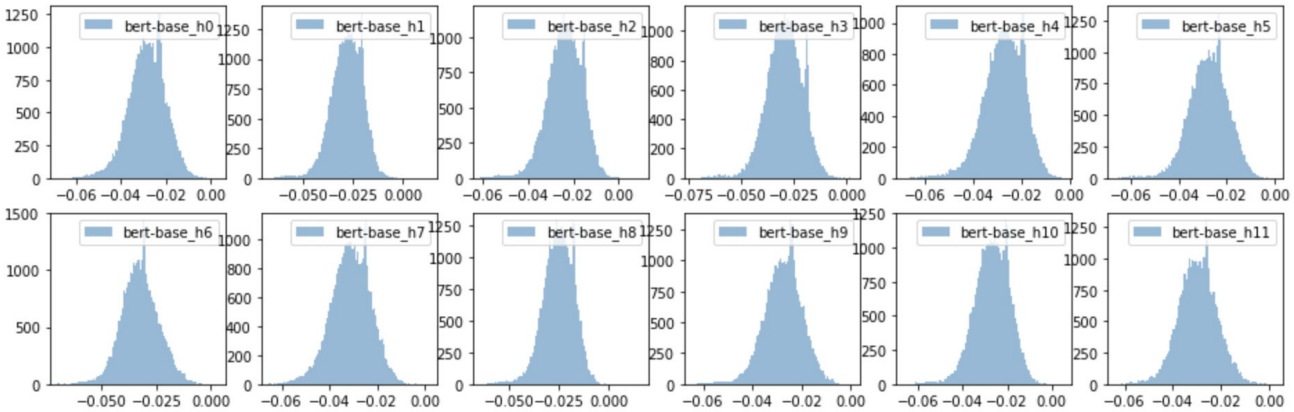
- **IWSLT14 De-En (0.16M translation pairs)**

- Why GN ?
  2. "Different" Gaussian for each head dim.

assume each feature dim x_i is a Gaussian ~ N(0, 1), average $\{x_1, \ldots, x_i\}$ will be N(0, 1))



Averaged embedding value of BERT-base for each head



Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

Small (36.7M Param) L6+L6 D512 H4	**BLEU**
Layer Norm + Deep init[2]	35.63
Layer Norm	34.95
Group Norm	**35.27**
Batch Norm	diverge
Mask Batch Norm	34.51 (w/ rm)[1] 33.79
Mask Instance Norm	33.89

Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Group Norm	**36.05**
Batch Norm	diverge
Mask Batch Norm	35.48 (w/ rm) 34.38
Mask Instance Norm	34,51

[1] w/ rm: using the running mean and running variance while doing the inference.
[2] Biao Zhang, Ivan Titov, Rico Sennrich, Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention, ACL'19

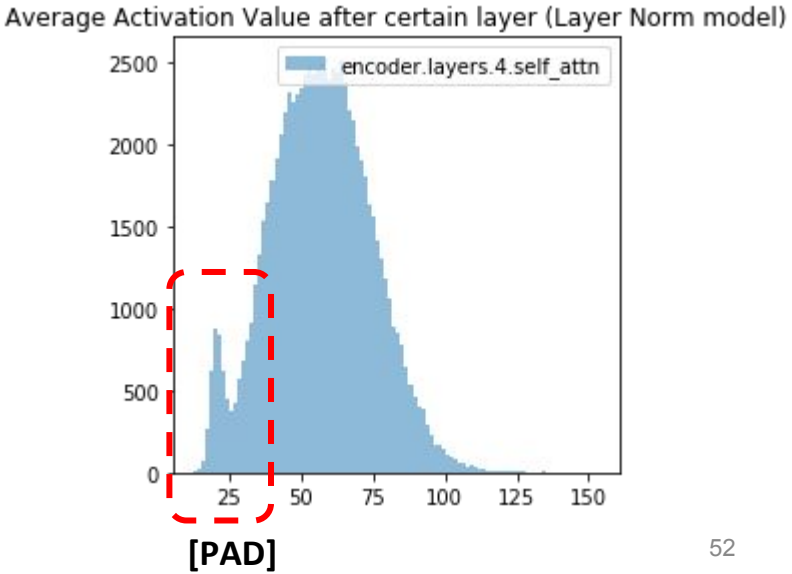
Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not vanilla BN ?
  1. Padding directly destroys the statistics.
     (10% of tokens per batch)

"the artist sold the painting." (5 tokens)

"Zhewei is cool." (3 token + 2 pad)



Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

Small (36.7M Param) L6+L6 D512 H4	**BLEU**
Layer Norm + Deep init[2]	35.63
Layer Norm	34.95
Group Norm	**35.27**
Batch Norm	diverge
Mask Batch Norm	34.51 (w/ rm)[1] 33.79
Mask Instance Norm	(w/o grad clip) nan 33.89

Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Group Norm	**36.05**
Batch Norm	diverge
Mask Batch Norm	35.48 (w/ rm) 34.38
Mask Instance Norm	(w/o grad clip) nan 34.51

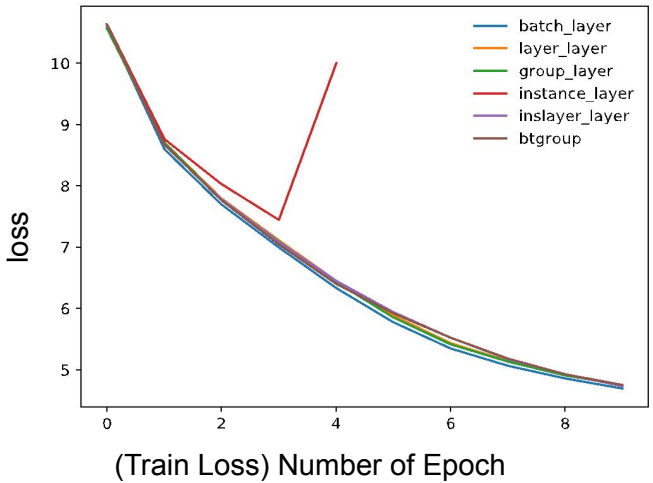
[1] w/ rm: using the running mean and running variance while doing the inference.
[2] Biao Zhang, Ivan Titov, Rico Sennrich, Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention, ACL'19

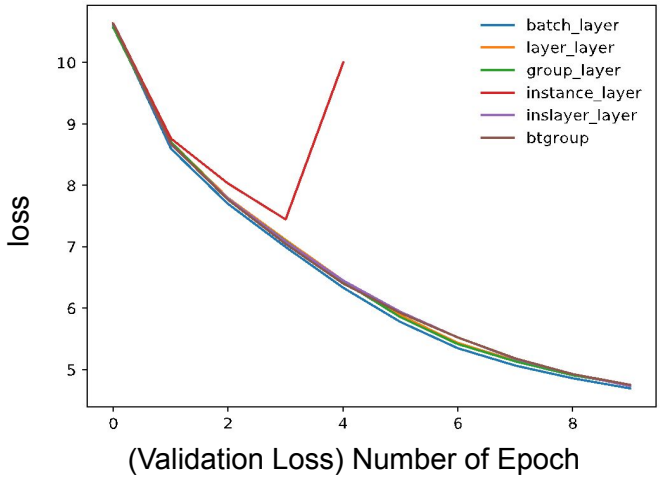
Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not IN ?

  1. Fail to control the scale of softmax. => gradient exploding.





Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ?
  1. Do we go deep enough.

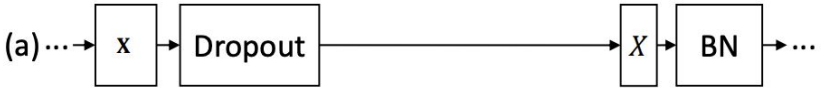
Deep (99.7M Param) L18+L6 D512 H4	**BLEU**	**Val loss**
Layer Norm	32.89	4.10
Mask Batch Norm	**34.3**	3.93

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ?

1. Dropout (0.3) destroys the variance during inference.
   2 * (a)-like structure in pre-norm transformer.



Skip Math: for each neuron x_i, assume $E[x_i] = c$, $Var[x] = v$, dropout keep ratio p.

$Var_{test}(x) / Var_{train}(x) = p * v / ((c^2 + v) - p*c^2)$

Base (68.2M Param) L12+L6 D512 H4	BLEU	diff
Layer Norm	35.77	
Mask Batch Norm	35.48 (w/ rm) 33.75	-0.29 -2.02
Layer Norm (w/o dp)	33.78	
Mask Batch Norm (w/o dp)	33.5 (w/ rm) 32.35	-0.28 -1.43

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ? => 1. Fix Dropout

  1.1 Residual-like dropout:

  Train: x = x + x * dp, dp ~ U(-b, b), Test: x

Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Mask Batch Norm	35.48 (w/ rm) 33.75
Mask Batch Norm	(w/ rm) 34.31

Skip Math: for each neuron x_i, assume $E[x_i] = c$, $Var[x] = v$, dropout keep ratio p.

$Var_{test}(x) / Var_{train}(x) \tilde{=} 3 / (3 + b^2)$

[1] Xiang Li, Shuo Chen, Xiaolin Hu, Jian Yang, CVPR'19, Understanding the Disharmony between Dropout and Batch Normalization by Variance Shift. (Momenta & Tsinghua)

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ? => 1. Fix Dropout

  1.2 Consistently correct the statistics while training.

  r = estimate_real_std / running_std,
  d = (estimate_real_mean - running_mean) / running_std
  X = (X - real_mean) / real_std * r + d

We use estimate_real_std instead of real_std for better correction.

Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Mask Batch Norm	35.48 (w/ rm) 33.75
Batch re-Norm	(w/ rm) **35.74**

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ? => 1. Fix Dropout

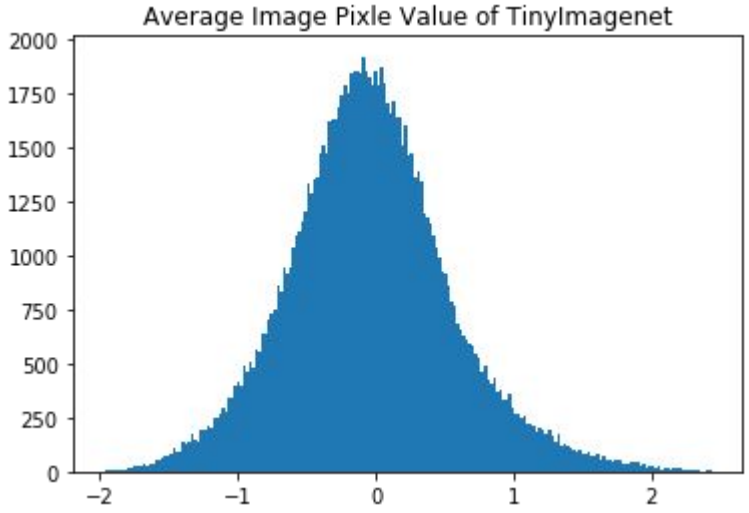
  1.3 Fuse the dropout and correct the running mean/variance directly.

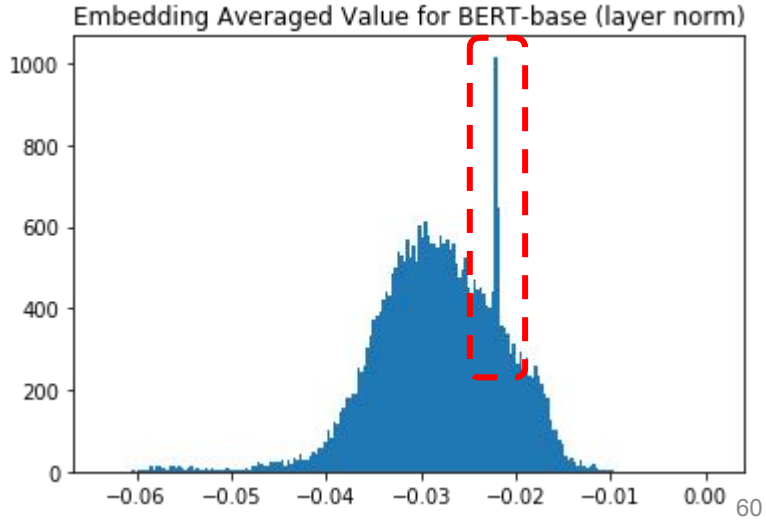
Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Mask Batch Norm	35.48 (w/ rm) 35.53

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ?
  2. Maybe not Gaussian for input.
     (suppose each feature dim x_i is a N(0, 1), Average $\{x_1, \ldots, x_i\}$ will be N(0, 1))

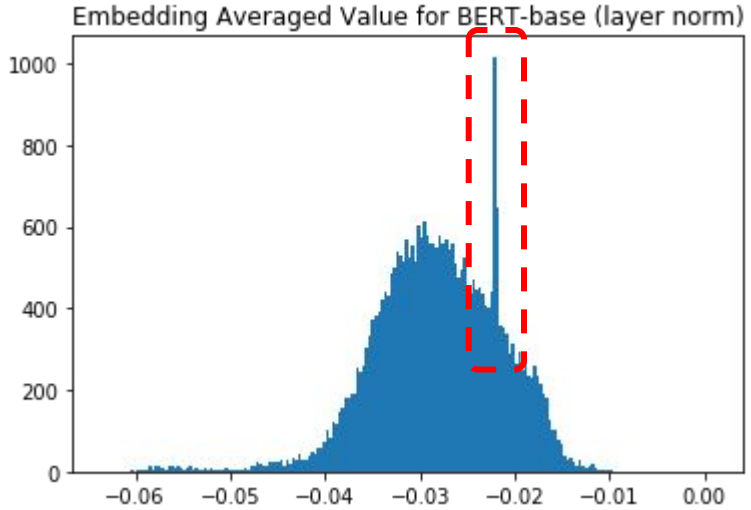




Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ?
  2. Maybe not Gaussian for input.
     (infrequent words will end up close in terms of distribution after optimization.[1])



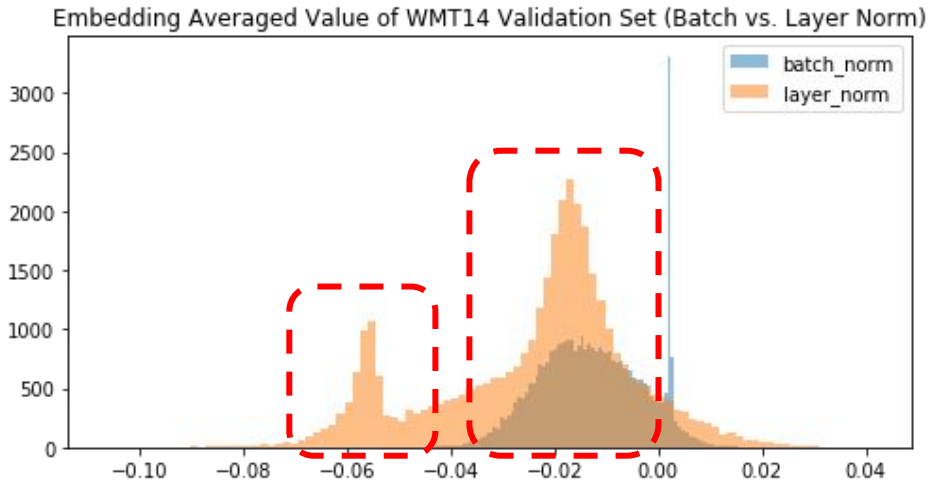
Infrequent words: (3% of the whole vocab)
Eg: granny, witches, algebra, linux, ...

[1] Gao et al. “Representation Degeneration Problem in Training Natural Language Generation Models,” ICLR’19

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

- Why not BN ?
  2. Maybe not Gaussian for input. (En & De)



Base (68.2M Param) L12+L6 D512 H4	**BLEU**
Layer Norm	35.77
Mask Batch Norm	35.48 (w/ rm) 33.75
Multi-mode Batch Norm (2-mode)	**35.58** (w/ rm) **34.38**

Normalization Results on Neural Machine Translation

- **IWSLT14 De-En (0.16M translation pairs)**

Base (68.2M Param) L12+L6 D512 H4	**BLEU**	**Val loss**
Layer Norm + Deep init[1]	35.63	-
Layer Norm	35.77	3.85
Group Norm	**36.05**	**3.83**
Mask Batch Norm	35.48 (w/ rm) 33.75	3.88
Multi-mode Batch Norm	35.58 (w/ rm) 34.38	3.87
Batch re-Norm	(w/ rm) 35.74	3.89
Instance-Layer Norm	35.68	3.87

[1] Zhang et al, Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention, ACL'19

Normalization Results on Neural Machine Translation

- **WMT14 En-De (4.5M translation pairs)**

Big (68.2M Param) L6+L6 D1024 H16	**BLEU**
Layer Norm	28.92
Group Norm	**29.12**

Base (36.7M Param) L6+L6 D512 H8	**BLEU**
Layer Norm + Deep init[1]	27.56
Scale-norm[2]	27.57
Layer Norm	27.86
Group Norm	**28.33**
Mask Batch Norm	26.53 (w/ rm) 24.93
Multi-mode Batch Norm	-
Batch re-Norm	28.0
Instance-Layer Norm	27.9

[1] Zhang et al, Improving Deep Transformer with Depth-Scaled Initialization and Merged Attention, ACL'19
[2] Nguyen et al, Transformers without Tears: Improving the Normalization of Self-Attention, IWSLT'19

Normalization Results on ROBERTA

- **Wiki + Book Corpus (213M sentences)**

Small (6.6M Param) L12 D128 H4 / Val loss bsz: 2k	Layer Norm	Group Norm	Mask Batch Norm (w/o running mean)	Multi-mode Batch Norm (w/o running mean)
10k	4.60	4.67	4.51	**4.48**
20k	4.21	4.29	4.17	**4.14**
40k	4.04	4.09	4.01	**3.99**
60k	3.96	3.99	3.93	**3.92**
80k	3.90	3.92	3.86	**3.84**
100k	3.85	3.87	3.82	**3.81**

Normalization Results on ROBERTA

- **Wiki + Book Corpus (213M sentences)**

Base (110M Param) L12 D768 H12 / Val loss bsz: 2k	Layer Norm	Group Norm	Mask Batch Norm (w/o running mean)	Multi-mode Batch Norm (w/o running mean)
10k	3.01	3.03	-	-
30k	2.55	2.52	-	-
50k	2.39	2.37	-	-
70k	2.29	2.28	-	-
90k	2.22	2.20	-	-
100k	2.18	2.17	-	-

Normalization Results on ROBERTA

- **Wiki + Book Corpus (213M sentences)**

Base (110M Param) L12 D768 H12 / Val loss bsz: 2k	MNLI (393k)	QNLI (105k)	QQP (364k)	SST-2 (67k)	RTE* (2.5k)	MRPC* (3.7k)	mean*
BERT-LN	0.843	0.893	0.895	0.917	0.713	0.886	0.8578333333
RoBERTA-LN	0.876	0.928	0.919	0.948	0.787	0.902	0.8933333333
Our-LN-base	0.849984	0.914466	0.91072	0.896205	0.725	0.896635	0.870896
Our-GN-base	0.852968	0.91659	0.91198	0.929888	0.73	0.894231	0.8726095

Content

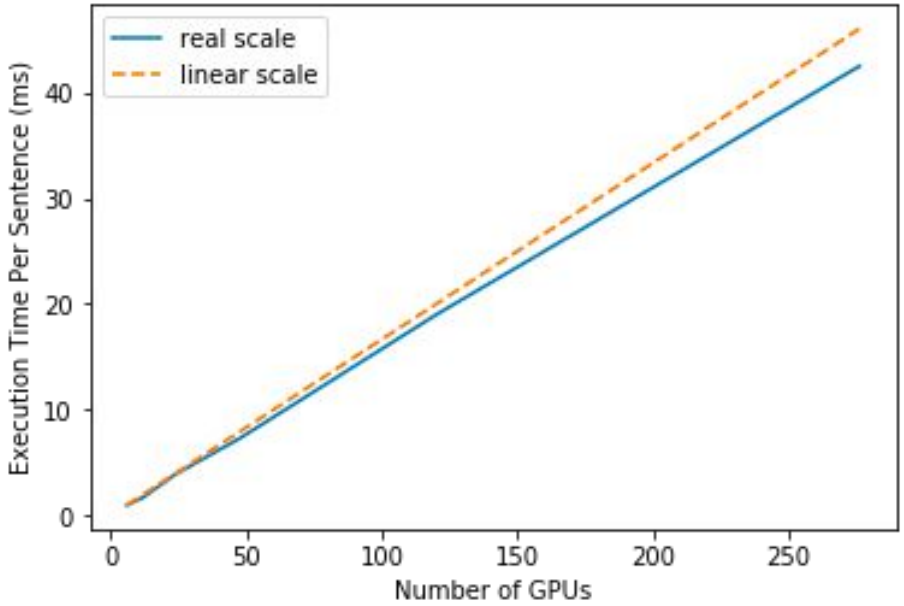
- How Normalization is performed in NLP (vs. CV)?

- Normalization results for different NLP tasks.

- **Other updates: scalability of ROBERTA training on summit.**

Scalability Test of ROBERTA training

- **RoBERTA-base:** (12 sentence per GPU. Averaged based on 30min runs.)



Conclusion

- Group Normalization performs well for MT.
- Exploring multi-mode batch normalization for Language Model-like tasks.